UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
10/659,066	09/09/2003	Brad Calder	0321.68199	3820

24978 7590 01/29/2007
GREER, BURNS & CRAIN
300 S WACKER DR
25TH FLOOR
CHICAGO, IL 60606

EXAMINER	
STEELMAN, MARY J	
ART UNIT	PAPER NUMBER
2191	

SHORTENED STATUTORY PERIOD OF RESPONSE	MAIL DATE	DELIVERY MODE
3 MONTHS	01/29/2007	PAPER

Please find below and/or attached an Office communication concerning this application or proceeding.

If NO period for reply is specified above, the maximum statutory period will apply and will expire 6 MONTHS from the mailing date of this communication.

Office Action Summary	Application No.	Applicant(s)	
	10/659,066	CALDER ET AL.	
	Examiner	Art Unit	
	Mary J. Steelman	2191	

-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address --

Period for Reply

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE 3 MONTH(S) OR THIRTY (30) DAYS, WHICHEVER IS LONGER, FROM THE MAILING DATE OF THIS COMMUNICATION.

- Extensions of time may be available under the provisions of 37 CFR 1.136(a). In no event, however, may a reply be timely filed after SIX (6) MONTHS from the mailing date of this communication.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133). Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any earned patent term adjustment. See 37 CFR 1.704(b).

Status

1)☒ Responsive to communication(s) filed on 30 October 2006.

2a)☒ This action is **FINAL**. 2b)☐ This action is non-final.

3)☐ Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

Disposition of Claims

4)☒ Claim(s) 1-53 is/are pending in the application.

4a) Of the above claim(s) _____ is/are withdrawn from consideration.

5)☐ Claim(s) _____ is/are allowed.

6)☒ Claim(s) 1-53 is/are rejected.

7)☐ Claim(s) _____ is/are objected to.

8)☐ Claim(s) _____ are subject to restriction and/or election requirement.

Application Papers

9)☐ The specification is objected to by the Examiner.

10)☐ The drawing(s) filed on _____ is/are: a)☐ accepted or b)☐ objected to by the Examiner.

Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).

Replacement drawing sheet(s) including the correction is required if the drawing(s) is objected to. See 37 CFR 1.121(d).

11)☐ The oath or declaration is objected to by the Examiner. Note the attached Office Action or form PTO-152.

Priority under 35 U.S.C. § 119

12)☐ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).

a)☐ All b)☐ Some * c)☐ None of:

1.☐ Certified copies of the priority documents have been received.

2.☐ Certified copies of the priority documents have been received in Application No. _____.

3.☐ Copies of the certified copies of the priority documents have been received in this National Stage application from the International Bureau (PCT Rule 17.2(a)).

* See the attached detailed Office action for a list of the certified copies not received.

Attachment(s)

1)☒ Notice of References Cited (PTO-892)

2)☐ Notice of Draftsperson's Patent Drawing Review (PTO-948)

3)☐ Information Disclosure Statement(s) (PTO/SB/08) Paper No(s)/Mail Date _____.

4)☐ Interview Summary (PTO-413) Paper No(s)/Mail Date. _____ .

5)☐ Notice of Informal Patent Application.

6)☐ Other: _____.

DETAILED ACTION

1. This Office Action is in response to Claim Amendments, Remarks and Specification Amendments received 10/30/2006. Per Applicant's request, claims 1, 12, 27, 31, 41, 42, 45, and 47 have been amended. Per Applicant's request, the Specification has been amended. Claims 1-53 are pending.

Specification

2. Examiner is unclear as to the intent to amend the Specification at page 8, line 4 and page 36, line 9. It appears that Specification was amended with the identical term.

Claim Rejections - 35 USC § 112

3. The following is a quotation of the second paragraph of 35 U.S.C. 112:

> The specification shall conclude with one or more claims particularly pointing out and distinctly claiming the subject matter which the applicant regards as his invention.

4. Examiner maintains the prior rejection of Claims 12 (and dependent claims 13-21 & 26), 41, 42 and 47 (and dependent claims 48-53). Claims 12 (and dependent claims 13-21 & 26), 41, 42 and 47 (and dependent claims 48-53) are rejected under 35 U.S.C. 112, second paragraph, as being indefinite for failing to particularly point out and distinctly claim the subject matter which applicant regards as the invention.

5. The term "more likely to be similar" in claim 12 is a relative term which renders the claim indefinite. The term "more likely to be similar" is not defined by the claim, the specification does not provide a standard for ascertaining the requisite degree, and one of ordinary skill in the art would not be reasonably apprised of the scope of the invention. A suggested amendment: "...using tracked statistic to classify the plurality of intervals of execution into at least one cluster, said one or more clusters based on similarity of behavior."

6. In view of the amendment to claim 41, the prior 35 U.S.C. 112 2nd paragraph rejection is hereby withdrawn.

7. In view of the amendment to claim 42, the prior 35 U.S.C. 112 2nd paragraph rejection is hereby withdrawn.

8. In view of the amendment to claim 47, the prior 35 U.S.C. 112 2nd paragraph rejection is hereby withdrawn.

Claim Rejections - 35 USC § 101

9. Regarding the 35 U.S.C. 101 rejection of Claims 1-43, Examiner has reconsidered and finds "running code" (executing code) to fall into a statutory category. The prior 35 USC 101 rejections are hereby withdrawn.

Response to Arguments

10. Applicant has argued, in substance, the following:

(A) As Applicant has noted on page 24, last paragraph, of Remarks, Goodnow neither teaches nor suggests at least running code of a computer program over an plurality of intervals of execution, identifying a behavior of the computer program over each of the plurality of intervals of execution, and comparing at least one identified behavior for at least one interval of execution to another interval of execution to determine similarity between the intervals of execution."

Applicant further points to the Specification, page 11, line 6, for a definition of interval: "a selection of continuous instructions in program execution order, which may include, for

example, a time interval, an instruction interval, and/or a metric-based interval." Applicant proposes (page 25, 3rd paragraph) that "the claimed interval is different from the code segment in Goodnow at least because an interval can include several code segments, repeat code segments, etc."

Examiner's Response:

Examiner disagrees. Goodnow disclosed (col. 8: 62-63), "the program is run (running code) so that the dynamic attributes may be extracted (step 215)", col. 10: 11-25, "The data structure identifies each function (a plurality of intervals) which is executed and determines the properties of the other function which will be executed...Each row and column represents a particular function. A function corresponds to a particular collection of blocks. The number identified in each space indicated the probability that a given function will transition to another given function..." Running code, identifying each function which is executed is consistent with a plurality of intervals. Claim language does not recite 'repeat code segments', however is should be noted that it is not uncommon for function code to be repeatedly executed. In fact, page 11, line 6, of the Specification defines 'interval' as "a selection of continuous instructions in program execution order." Examiner finds executing a function, and determining the properties of the function, to be analogous. Additionally, Table 6 (col. 9) depicts behavior of basic blocks, relating the frequency that the block repeats itself, or the frequency that the block calls another block (behavior statistics). Col. 11, lines 50-54, "degree of similarity between the identifiers and function types contained within the two code segments being compared (determine similarity between the intervals of execution)."

Goodnow disclosed (col. 4, lines 39-41), "Similarity measurements are performed between two or more code segments to identify how closely related the two code segments are (comparing at least one identified behavior / determine similarity) for some defined set of properties."

(B) As Applicant has noted on page 25, 2nd paragraph, Goodnow does not address the case of 'code segments that execute within a particular interval...'

Examiner's Response:

'Within a particular interval' is not a claim limitation. In response to applicant's argument that the references fail to show certain features of applicant's invention, it is noted that the features upon which applicant relies (i.e., 'within a particular interval') is not recited in the rejected claim(s). Although the claims are interpreted in light of the specification, limitations from the specification are not read into the claims. See *In re Van Geuns*, 988 F.2d 1181, 26 USPQ2d 1057 (Fed. Cir. 1993).

(C) Regarding independent claim 27, as noted on page 25, last paragraph, Goodnow fails to teach or suggest at least identifying behavior of a hardware-independent metric within at least one arbitrary section of execution of a portion of a computer program, or classifying each of the at least one arbitrary section of execution according to the identified behavior into clusters of similar behavior.

Applicant argues, "Goodnow identifies behavior of code segments, not behavior of arbitrary sections of program execution."

Examiner's Response:

Examiner fails to see the difference between comparing the behavior of various code segments, such as shown in Tables 6-9, and 'arbitrary sections of program execution'. Table 6 shows a comparison of Block #s, with behavior (which block is next executed). Block 1 is 'arbitrarily' compared to next executing Block 1, Block 2, etc.

(D) Regarding independent claim 28, as noted on page 26, first paragraph, "cited portion refers to occurrences of an attribute within a block, not frequency of basic blocks of executed within an arbitrary section of program execution."

Examiner's Response:

Examiner points to Table 6, col. 9, lines 17-19, "the number 14 in the first space indicates that block number 1 will probably transition to itself fourteen times (frequency of basic blocks of executed within an arbitrary section of program execution)."

(E) Regarding independent claim 35, as noted on page 26, second paragraph, "Goodnow fails to teach or suggest at least identifying behavior of a hardware-independent metric for each of a plurality of intervals of execution, comparing the identified behavior of each of the plurality of intervals to the identified target behavior over full execution to determine a representative

interval, or simulating execution of the computer program over the determined representative interval." "Goodnow compares code segments, not intervals of execution.

Applicant argues that limitations call for 'simulating execution", but the Goodnow reference merely teaches 'running the program and extracting dynamic attributes."

Applicant argues that there is no citation for 'determining a representative interval." (determined representative interval)

Examiner's Response:

See Table 6, where Basic Block behavior for a plurality of Basic Blocks is shown (identifying behavior of a hardware-independent metric for each of a plurality of intervals of execution). The columns represent the Basic Block to next be executed (hardware independent metric).

See col. 13, lines 10-53 regarding a discussion of the cluster interface representing the similarity of the code segments being compared. (comparing the identified behavior of each of the plurality of intervals to the identified target behavior over full execution to determine a representative interval) See col. 4, lines 41-46, "The code segments may be identified (determine a representative interval) as two functions within either the same program or different programs or any other defined designation such as, but not limited to blocks, Lvalues or statements." Note that patent application Ser. No. 08/373,340 (USPN 5,574,837) is incorporated

by reference (col. 13, line 13). USPN 5,574,837 provides additional details on a cluster tree generator (col. 2, lines 33-47).

See (col. 8: 62-63), "the program is run (simulating execution) so that the dynamic attributes may be extracted (step 215)", col. 10: 11-25, "The data structure identifies each function (a plurality of intervals) which is executed and determines the properties of the other function which will be executed...Each row and column represents a particular function. A function corresponds to a particular collection of blocks. The number identified in each space indicated the probability that a given function will transition to another given function..." Col. 11, lines 24-30, "The dynamic data provides an indication...The use of the data is determined by observing..." (simulating execution). Running the program and extracting statistical information is analogous to 'simulating execution.' (See article attached titled "What is Simulation?" for a definition of simulation. A declarative, functional model is executed and analyzed. Essentially the article equates 'simulation' and 'execution' analysis of defined elements.

Running code, identifying each function which is executed is consistent with a plurality of intervals. Claim language does not recite 'repeat code segments', however is should be noted that it is not uncommon for function code to be repeatedly executed. In fact, page 11, line 6, of the Specification defines 'interval' as "a selection of continuous instructions in program execution order." Examiner finds executing a function, and determining the properties of the function, to be analogous. Additionally, Table 6 (col. 9) depicts behavior of basic blocks, relating the frequency that the block repeats itself, or the frequency that the block calls another

block (behavior statistics). Col. 11, lines 50-54, "degree of similarity between the identifiers and function types contained within the two code segments being compared (determine similarity between the intervals of execution)."

Claim Rejections - 35 USC § 102

11. The following is a quotation of the appropriate paragraphs of 35 U.S.C. 102 that form the basis for the rejections under this section made in this Office action:

> A person shall be entitled to a patent unless –
>
> (b) the invention was patented or described in a printed publication in this or a foreign country or in public use or on sale in this country, more than one year prior to the date of application for patent in the United States.

12. Claims 1-17, 22-33, 35-40, 42-48, 50, and 52 are rejected under 35 U.S.C. 102(b) as being anticipated by USPN 5,699,507 to Goodnow, II et al.

Per claim 1:

A method of analyzing a computer program, the method comprising:

-running code of the computer program over a plurality of intervals of execution;

Goodnow: Col. 8: 62-63, "the program is run so that the dynamic attributes may be extracted (step 215)", col. 10: 11-25, "The data structure identifies each function (a plurality of intervals) which is executed and determines the properties of the other function which will be executed...Each row and column represents a particular function. A function corresponds to a particular collection of blocks. The number identified in each space indicated the probability that a given function will transition to another given function..."

-during said step of running code, tracking a statistic for a program component;

Goodnow: Col. 8, lines 63-66, "the program is run so that the dynamic attributes may be extracted (step 215)....may include...control flow and data flow analysis."

-identifying a behavior of the computer program over each of the plurality of intervals of execution based on the tracked statistic;

Goodnow: Col. 10, see Table 7 and text at col. 10: 11-25, "The data structure identifies each function (plurality of intervals) which is executed and determines the properties of the other function which will be executed...probability that a given function will transition (behavior)..."

-comparing at least one identified behavior for at least one interval of execution to another interval of execution to determine similarity between the intervals of execution.

Goodnow: Table 6 & Col. 9: 11-26, "This data structure identifies each block (of an interval / function) which is executed and determines the properties of the other block which will be executed...indicated the probability that a given block will transition...It is to be understood by those skilled in the art that the above data structure b_dynamic.ctrl is an exemplary history matrix which may be used to represent the data and that the data can be formatted in various ways..." Col. 4, lines 39-41, "Similarity measurements are performed between two or more code segments to identify how closely related the two code segments are for some defined set of properties (identified behavior)." Col. 11, lines 50-54, "The distance function produces a distance measurement which indicates the degree of similarity between the identifiers and

function types contained within the two code segments being compared (determine similarity between the intervals of execution)."

Per claim 2:

-at least one of executing the program on hardware, simulating the program's execution in software, direct execution, emulating the program's execution in software, and modeling a hypothetical execution in software.

Goodnow: Col. 7: 29-32, "The types of operators identified in the program...were determined by observing how each of the operations is performed in hardware and then making a determination of which hardware operations are similar to one another."

Per claim 3:

-the statistic comprises at least one of a hardware metric and a hardware-independent metric.

Goodnow: Col. 7: 29-32.

Per claim 4:

-the statistic comprises at least one of frequency of the component occurring in execution, number of instructions executed, amount of memory used by the program component, time, IPC, performance counters, program counters, and cache miss rate.

Goodnow: Col. 3: 1-12, "information about variables, operators, control flow and data flow", col. 6: 39, "number of occurrences of each identifier types", col. 7: 20, "number of occurrences

of a given parameter", col. 8: 31, "number of occurrences of each operator", col. 8: 45-61, "number of occurrences of that particular identifier or operator in that function".

Per claim 5:

-the program component comprises an identifiable section of control flow of the computer program.

Goodnow: Col. 8: 64, "control flow", col. 3, 6-8, "Control flow information generally includes a history of every file name and line number which has been visited."

Per claim 6:

-the program component comprises at least one of an instruction, basic block, procedure, loop, load instruction, and branch instruction.

Goodnow: Col. 7: 26, col. 14: 63-64, Also see code sample at col. 3: 23.

Per claim 7:

-the program component comprises a memory region.

Goodnow: Col. 2: 37-39

Per claim 8:

-the program component comprises a basic block, the basic block being a section of the code having a single entry point and a single exit.

Goodnow: Col. 8:56, col. 9: 11-26, Also see code sample at col. 3: 23.

Per claim 9:

-each of the plurality of intervals of execution comprises at least one of a time interval, an instruction interval, and a metric-based interval.

Goodnow: Col. 3: 9-12- data flow information recorded, Col. 10: 12-25,

Per claim 10:

-one of the plurality of intervals comprises the execution of at least one of the subset of the code and the full execution of the code.

Goodnow: Col. 8: 62-63- entire program is executed, including all function intervals.

Per claim 11:

-the intervals of execution comprise at least one of overlapping and non-overlapping intervals.

Goodnow: Col. 9: 28-30, Table 7 & col. 10: 12-26

Per claim 12:

-based on said comparing step, classifying the plurality of intervals of execution into at least one cluster,

Goodnow: Col. 11: 30-39.

-wherein each of the intervals is more likely to be similar in program behavior to the other intervals in its cluster than to the intervals in a remainder of clusters.

Goodnow: col. 11: 50-64. Using the matrix tables the data points are clustered in an interface.

Per claim 13:

-selecting at least one representative interval of execution for each of the at least one cluster.

Goodnow: Col. 13: 8-12, FIG. 3, #325, Col. 13: 37,

Per claim 14:

-each of the at least one representative interval of execution is closest to an average behavior of the cluster.

Goodnow: Col. 13: 48-50, "distance measurement and indicate the degree of similarity relative to the other code segments…", col. 4: 39-41, "Similarity measurements are performed between two or more code segments to identify how closely related the two code segments are for some defined set of properties."

Per claim 15:

-the representative interval of execution is the earliest interval of execution within a predetermined distance from an average behavior of the cluster.

Goodnow: Col. 3: 5-12.

Per claim 16:

-weighing each of the selected representative intervals of execution based on at least one of a total amount of time, a number of instructions within the cluster, the program component, and the statistic.

Goodnow: Col. 13: 17-36.

Per claim 17:

-the weighted representative intervals collectively represent a complete execution of at least a subset of the computer program.

Goodnow: Col. 9: 11, col. 10: 12, col. 8: 62-63.

Per claim 22:

-comparing each interval to the interval of execution representing at least a subset of execution of the computer program.

Goodnow: Col. 11: 31-36.

Per claim 23:

-based on said comparing step, identifying an end of an initialization of the computer program.

Goodnow: Col. 11: 61-64, col. 5: 16, col. 8: 61-62.

Per claim 24:

-based on said comparing step, identifying a length of at least one repeating interval of

execution.

Goodnow: Col 12: 49.

Per claim 25:

-identifying a length comprises performing an analysis of a signal, the signal comprising differences between each identified interval of execution and the interval of execution representing the at least a subset of execution of the computer program.

Goodnow: Col. 11: 54-58.

Per claim 26:

-determining a confidence and variance by sampling the intervals of execution within a particular cluster for at least one of a hardware metric and a hardware-independent metric.

Goodnow disclosed (Col. 2: 48-51, 58-61 & 65-66) a distance matrix generator 115 receives the information from the statistical generator 10 and identifies similarities…as a function of a weighing scheme (confidence and variance)…,an interface generator generates an interface such as, but not limited to, a cluster interface or dot plot matrix, which is based on the generated distance matrices. Predefined metrics which include both static and dynamic characteristics are considered in identifying similarities in two or more code segments.

Per claim 27:

A method of analyzing a computer program, the method comprising:

-running at least a portion of the computer program;

Goodnow: Col. 8: 62.

-identifying behavior of a hardware-independent metric within at least one arbitrary section of execution of the portion of the computer program during said running step;

Goodnow: Col. 8: 62-66.

-classifying each of the at least one arbitrary section of execution according to the identified behavior into clusters of similar behavior.

Goodnow: Col. 9: 24-26 – history matrix is used in generating a (col. 2: 60) cluster interface / dot plot matrix.

Per claim 28:

-identifying a frequency of execution of basic blocks of the executed code,

Goodnow: Col. 8: 55-56. Also see, Table 6, col. 9, lines 17-19, "the number 14 in the first space indicates that block number 1 will probably transition to itself fourteen times (frequency of basic blocks of executed within an arbitrary section of program execution)."

-wherein each of the at least one basic block comprises a piece of code of the computer program executed from start to finish, said basic block having only one entry point and one exit.

Goodnow: Col. 9: 11-26 Also see rejection of claim 8 above.

Per claim 29:

-identifying a frequency provides a group of frequencies for each of the number of intervals.

Goodnow: Col. 9: 24 (history matrix).

Per claim 30:

-comparing the identified behavior of one of the intervals to the identified behavior of another of the intervals to identify a phase of the interval.

Goodnow: Col. 9: 24-26 A history matrix identifies the behavior of the functions and a phase transition to another function. See Table 7 for function transitions.

Per claim 31:

-identifying an initialization phase;

Goodnow: See rejection of claim 23 above.

-determining at least one analysis point occurring after execution of the identified initialization phase.

Goodnow: Col. 13: 37-40 Compare function similarities.

Per claim 32:

-for each of the number of intervals, determining a interval vector, the interval vector comprising a plurality of ordered elements, each of the plurality of ordered elements relating to a particular

basic block and representing a frequency of execution of the particular basic block.

Goodnow: Col. 8: 53, col. 9: 11-26.

Per claim 33:

-partitioning the computer program into a set of clusters by comparing the determined interval vectors to one another.

Goodnow: Col. 9: 27 – create f_dynamic.ctrl & use data sets to create a cluster interface (col. 2: 60).

Per claim 35:

A method of analyzing operation of a computer program, the method comprising:

-executing at least a portion of the computer program;

-for each of a plurality of intervals of execution over the at least a portion of the computer program, identifying behavior of a hardware-independent metric;

-identifying behavior of the hardware-independent metric over full execution of the at least a portion of the computer program to identify a target behavior;

-comparing the identified behavior of each of the plurality of intervals to the identified target behavior over full execution to determine a representative interval;

-simulating execution of the computer program over the determined representative interval.

See Table 6, where Basic Block behavior for a plurality of Basic Blocks is shown (identifying behavior of a hardware-independent metric for each of a plurality of intervals of execution). The columns represent the Basic Block to next be executed (hardware independent metric).

See col. 9-col. 12. Basic blocks (portions of the computer program) are fully executed and analyzed for such details as transitions to a cited successive block (target behavior), identifying each time one of the identifiers is referenced (col. 10, line 28) (over full execution of the at least a portion of the computer program to identify a target behavior)

Col. 2, lines 41-42, "code segment source 105 may be a process which generates segments of code (determined representative interval) and sends the code segments to statistical extractor 110 in real time. See col. 13, lines 10-53 regarding a discussion of the cluster interface representing the similarity of the code segments being compared. (comparing the identified behavior of each of the plurality of intervals to the identified target behavior over full execution to determine a representative interval) See col. 4, lines 41-46, "The code segments may be identified (determine a representative interval) as two functions within either the same program or different programs or any other defined designation such as, but not limited to blocks, Lvalues or statements." Note that patent application Ser. No. 08/373,340 (USPN 5,574,837) is incorporated by reference (col. 13, line 13). USPN 5,574,837 provides additional details on a (col. 2, lines 24-48) "code segment source 105 may be a process which generates segments of code and sends the code segments to statistical extractor 110...The statistical extractor 110 extracts...from each code segment...a distance matrix generator 115 receives the statistics...and identifies similarities

between the segments of code...and a cluster tree generator (col. 2, line 43) generates cluster trees (representing the determined representative interval) based on the generated distance matrices."

See (col. 8: 62-63), "the program is run (simulating execution) so that the dynamic attributes may be extracted (step 215)", col. 10: 11-25, "The data structure identifies each function (a plurality of intervals) which is executed and determines the properties of the other function which will be executed...Each row and column represents a particular function. A function corresponds to a particular collection of blocks. The number identified in each space indicated the probability that a given function will transition to another given function..." Col. 11, lines 24-30, "The dynamic data provides an indication...The use of the data is determined by observing..." (simulating execution). Running the program and extracting statistical information is analogous to 'simulating execution.' (See article attached titled "What is Simulation?" for a definition of simulation. A declarative, functional model is executed and analyzed. Essentially the article equates 'simulation' and 'execution' analysis of defined elements.

The Specification defines 'interval' as "a selection of continuous instructions in program execution order." Examiner finds executing a function, and determining the properties of the function, to be analogous. Additionally, Table 6 (col. 9) depicts behavior of basic blocks, relating the frequency that the block repeats itself, or the frequency that the block calls another block (behavior statistics). Col. 11, lines 50-54, "degree of similarity between the identifiers and

function types contained within the two code segments being compared (determine similarity between the intervals of execution)."

Per claim 36:

-deriving a plurality of basic block vectors, each basic block vector representing code blocks of the program executed during the interval of execution, the basic block vector being based on frequencies of basic blocks of executed code within execution of the program;

-wherein the basic block vector comprises a single dimensional array where a single element in the array exists for a basic block in the program.

Goodnow: Col. 14: 63-64, col. 9: 11-13. See Table 6-"identifies each block which is executed..."

Per claim 37:

-the method is performed in run-time.

Goodnow: Col. 8: 62-67.

Per claim 38:

-identifying comprises tracking a proportion of instructions executed from different sections of code of the program over each of the plurality of intervals; further comprising, for each interval, classifying the identified behavior into phases corresponding to changes in behavior across the executed code.

Goodnow: See Tables 6 (basic blocks) & 7 (functions / phases). Table 7 shows (col. 10: 20) that a given function will transition to another given function (to another phase).

Per claim 39:

-predicting when execution of the code is about to enter a phase change; predicting a phase entered by the phase change.

Goodnow: Table 7 shows (col. 10: 20) that a given function will transition (about to enter) to another given function (to another phase).

Per claim 40:

-for each section of code: capturing an identifier of the section of code;

-capturing a number of instructions executed for the section of code.

Goodnow: See example at Table 6 (block number). Table 8 & related text at col. 11: 13-40, "indicates when an identifier has been referenced."

Per claim 42:

-comparing each section of code to a previous section of code in a history;

-if the compared section of code is not similar to the previous section of code, adding the previous section of code to the history.

Goodnow: Col. 9: 22-26 discloses a history matrix.. Col. 2: 42-43, "The statistical extractor 110 extracts predefined data attributes from each code segment", col. 11: 21 history.

Per claim 43:

-the behavior identified for an interval is collected in a vector, the vector containing the statistic for at least one element representing at least one component.

Goodnow: Col. 10: 17-21.

Per claim 44:

-the vector is retained in at least one of a memory, a storage medium, and a table.

Goodnow: See Table 7 at column 10.

Per claim 45:

-the vector is stored as a signature that represents at least one of the behavior of a complete vector, a projection of the vector, a compressed representation of the vector, a partial representation of the vector, and a subset of the identified behavior collected in the vector.

Goodnow: Table 7 Col. 10: 1-21 & Col. 11: 57.

Per claim 46:

-storing a phase ID with the signature, wherein the phase ID comprises at least one of a complete signature, a subset of the signature, a partial representation of the signature, a name independent of the signature, and a number.

Goodnow: Col. 11: 57 (signatures)

Per claim 47:

-the stored phase ID is identified for an interval by looking up the signature in storage, and based on said looking up, either using the phase ID stored with the signature, or creating a new phase ID.

Goodnow: See Tables 7 & 8. Table 7 shows functions transitioning to another given function (phases). Table 8 identifies identifiers modified or referenced.

Per claim 48:

-the identified behavior and the tracked statistic for at least one interval with a phase ID are stored and associated with one another.

Goodnow: See related Tables 7 & 8.

Per claim 50:

-after the phase ID is identified by a signature for an interval: looking up the phase ID to find the associated statistic for the interval.

Goodnow provided for (col. 2: 65-66) "metrics which include both static and dynamic characteristics are considered in identifying similarities in two or more code segments" using a dot plot matrix or cluster interface.

Per claim 52:

-the phase ID is stored in a prediction table,

-and further comprising: predicting a phase ID for an interval using the stored phase ID.

Goodnow: Col. 10: 19.

Claim Rejections - 35 USC § 103

13. The following is a quotation of 35 U.S.C. 103(a) which forms the basis for all obviousness rejections set forth in this Office action:

> (a) A patent may not be obtained though the invention is not identically disclosed or described as set forth in section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains. Patentability shall not be negatived by the manner in which the invention was made.

14. Claim 34 is rejected under 35 U.S.C. 103(a) as being unpatentable over USPN 5,699,507 to Goodnow, II et al., in view of USPN 5,574,837 to Clark et al.

Per claim 34:

Goodnow failed to explicitly disclose:

-determining a group of clusters;

-comparing each of the interval vectors to each of the set of clusters;

-adding the compared interval vector to a particular cluster based on a goodness of fit between the compared basic block vectors and each of the group of clusters;

-changing a centroid of each of the group of clusters; repeating the comparing, adding, and clustering steps to form the set of clusters.

However Clark more explicitly disclosed 'generating a browser interface for representing similarities between segments of code'. Col. 2: 25-29, "The statistical extractor 110 extracts predefined data attributes from each code segment…" Col. 2, lines 34-37, "A distance matrix

generator 115 receives the statistics from statistical extractor 110 and identifies similarities between the segments of code as a function of a weighting scheme…" Col. 2: 41-44, "scan the segments of code and select a specific weighting scheme based on criteria which defines the segments…Cluster tree generator 120 generates cluster trees based on the generated distance matrices"

Therefore, it would have been obvious, to one of ordinary skill in the art, at the time of the invention to modify Goodnow, using the teachings of Clark which include statistical functions performed on matrix data, because the Goodnow patent references the Clark patent (Goodnow: col. 13: 14-17). Both references disclose cluster interfaces representing similarities in segments of code.

15. Claims 18-21, 41, 49, 51, and 53 are rejected under 35 U.S.C. 103(a) as being unpatentable over USPN 5,699,507 to Goodnow, II et al., in view of USPN 5,574,837 to Clark et al., and further in view of USPN 5,953,006 to Baker et al.

Per claims 18 & 19:

Regarding the following limitations:

-minimizing the number of clusters.

Goodnow: Col. 2: 49-63, "A distance matrix generator 115 receives the information from the statistical generator 110 and identifies similarities between the segments of code as a function of a weighing scheme…the weighing scheme may be determined by a number of means, e.g.,

interaction with a user of the system 100 or application programs which scan the code segments and select a specific weighing scheme based on criteria which defines the segments...An interface generator 120 generates an interface such as...a cluster interface or dot plot matrix, which is based on the generated distance matrices.

Goodnow disclosed creating clusters using various weights & statistics, but failed to explicitly disclose minimizing the numbers of clusters.

Clark disclosed additional details related to using weights to emphasize statistical functions :

Col. 5: 40-63, "The distance functions are used to construct the cluster interface. FIG. 5 shows a display 500 which illustrates a cluster interface generated...maps the similarities...based on certain predefined constraints...The constraints are generally defined in terms of a weighting scheme which may be selected by the user to generate the cluster interface...The weights may place additional emphasis on certain structural features...used to emphasize certain functions..."

However, Baker more explicitly disclosed (col. 9: 6-14 & 35-44), compression and filtering to compress for storage of filter in the case of too many dots shown in a dot plot. Also see col. 4: 45-67 which discusses a dot plot views and a magnified view (using no compression).

Therefore, it would have been obvious, to one of ordinary skill in the art, at the time of the invention to modify Goodnow / Clark by including the more specific details related to filtering techniques related to compression, as disclosed by Baker because the Goodnow patent references the Clark patent (Goodnow: col. 13: 14-17) and the Clark patent, incorporates by reference the

Baker patent (Clark: col. 3: 20-24). All references disclose cluster interfaces representing similarities in segments of code.

Per claims 20 & 21:

-weighing each of the selected representative intervals of execution based on at least one of a total amount of time, a number of instructions within the cluster, the program component, and the statistic;

-wherein the weighted representative intervals collectively represent a complete execution of at least a subset of the computer program; and further comprising minimizing the number of clusters.

Goodnow: Col. 13: 17-36.

Goodnow failed to disclose 'minimizing.' See rejection of claims 18 and 19 above.

Therefore, it would have been obvious, to one of ordinary skill in the art, at the time of the invention to modify Goodnow / Clark by including the more specific details related to filtering techniques related to compression, as disclosed by Baker because the Goodnow patent references the Clark patent (Goodnow: col. 13: 14-17) and the Clark patent, incorporates by reference the Baker patent (Clark: col. 3: 20-24). All references disclose cluster interfaces representing similarities in segments of code.

Per claim 41:

-reducing a number of the identified sections of code to a lower number.

See rejection of claims 18 & 19 as related to minimizing / reducing.

Therefore, it would have been obvious, to one of ordinary skill in the art, at the time of the invention to modify Goodnow / Clark by including the more specific details related to filtering techniques related to compression, as disclosed by Baker because the Goodnow patent references the Clark patent (Goodnow: col. 13: 14-17) and the Clark patent, incorporates by reference the Baker patent (Clark: col. 3: 20-24). All references disclose cluster interfaces representing similarities in segments of code.

Per claim 49:

-if a storage area for storing the phase ID, behavior, and statistic is finite, only a single stored signature for a phase ID, and the phase ID, are stored.

See rejection of claims 18 & 19 as related to optimizing storage.

Therefore, it would have been obvious, to one of ordinary skill in the art, at the time of the invention to modify Goodnow / Clark by including the more specific details related to filtering techniques related to compression, as disclosed by Baker because the Goodnow patent references the Clark patent (Goodnow: col. 13: 14-17) and the Clark patent, incorporates by reference the Baker patent (Clark: col. 3: 20-24). All references disclose cluster interfaces representing similarities in segments of code.

Per claim 51:

-using the found associated statistic, performing at least one of a behavior optimization, statistic optimization, load-time optimization, run-time optimization, and hardware reconfiguration.

See rejection of claims 18 & 19 as related to optimizing.

Therefore, it would have been obvious, to one of ordinary skill in the art, at the time of the invention to modify Goodnow / Clark by including the more specific details related to filtering techniques related to compression, as disclosed by Baker because the Goodnow patent references the Clark patent (Goodnow: col. 13: 14-17) and the Clark patent, incorporates by reference the Baker patent (Clark: col. 3: 20-24). All references disclose cluster interfaces representing similarities in segments of code.

Per claim 53:

Regarding the limitations:

-retrieving information for the predicted phase ID;

-using the retrieved information, guiding optimization for the computer program.

Goodnow provided for (col. 2: 65-66) "metrics which include both static and dynamic characteristics are considered in identifying similarities in two or more code segments" using a dot plot matrix or cluster interface.

Goodnow failed to explicitly disclose optimization. See rejection of claims 18 & 19 as related to optimizing.

Therefore, it would have been obvious, to one of ordinary skill in the art, at the time of the invention to modify Goodnow / Clark by including the more specific details related to filtering techniques related to compression, as disclosed by Baker because the Goodnow patent references the Clark patent (Goodnow: col. 13: 14-17) and the Clark patent, incorporates by reference the Baker patent (Clark: col. 3: 20-24). All references disclose cluster interfaces representing similarities in segments of code.

Conclusion

16. Applicant's amendment necessitated the new ground(s) of rejection presented in this Office action. Accordingly, **THIS ACTION IS MADE FINAL**. See MPEP § 706.07(a). Applicant is reminded of the extension of time policy as set forth in 37 CFR 1.136(a).

A shortened statutory period for reply to this final action is set to expire THREE MONTHS from the mailing date of this action. In the event a first reply is filed within TWO MONTHS of the mailing date of this final action and the advisory action is not mailed until after the end of the THREE-MONTH shortened statutory period, then the shortened statutory period will expire on the date the advisory action is mailed, and any extension fee pursuant to 37 CFR 1.136(a) will be calculated from the mailing date of the advisory action. In no event, however, will the statutory period for reply expire later than SIX MONTHS from the date of this final action.

17. The prior art made of record and not relied upon is considered pertinent to applicant's disclosure.

Any inquiry concerning this communication or earlier communications from the examiner should be directed to Mary Steelman, whose telephone number is (571) 272-3704. The examiner can normally be reached Monday through Thursday, from 7:00 AM to 5:30 PM If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, Wei Zhen can be reached at (571) 272-3708. The fax phone number for the organization where this application or proceeding is assigned: 571-273-8300.

Any inquiry of a general nature or relating to the status of this application should be directed to the TC 2100 Group receptionist: 571-272-2100.

Information regarding the status of an application may be obtained from the Patent Application Information Retrieval (PAIR) system. Status information for published applications may be obtained from either Private PAIR or Public PAIR. Status information for unpublished applications is available through Private PAIR only. For more information about the PAIR system, see http://pair-direct.uspto.gov. Should you have questions on access to the Private PAIR system, contact the Electronic Business Center (EBC) at 866-217-9197 (toll-free).

Mary Steelman
01/16/2007

Mary Steelman
Primary Examiner
1-23-2007